B 10

```
PPPPPPPPPPPP     HHH         HHH     000000000     NNN         NNN  EEEEEEEEEEEEEEEE
PPPPPPPPPPPP     HHH         HHH     000000000     NNN         NNN  EEEEEEEEEEEEEEEE
PPPPPPPPPPPP     HHH         HHH     000000000     NNN         NNN  EEEEEEEEEEEEEEEE
PPP         PPP  HHH         HHH  000         000  NNN         NNN  EEE
PPP         PPP  HHH         HHH  000         000  NNN         NNN  EEE
PPP         PPP  HHH         HHH  000         000  NNN         NNN  EEE
PPP         PPP  HHH         HHH  000         000  NNNNNN      NNN  EEE
PPP         PPP  HHH         HHH  000         000  NNNNNN      NNN  EEE
PPP         PPP  HHH         HHH  000         000  NNNNNN      NNN  EEE
PPPPPPPPPPPP     HHHHHHHHHHHHHHH  000         000  NNN   NNN   NNN  EEEEEEEEEEEEE
PPPPPPPPPPPP     HHHHHHHHHHHHHHH  000         000  NNN   NNN   NNN  EEEEEEEEEEEEE
PPPPPPPPPPPP     HHHHHHHHHHHHHHH  000         000  NNN   NNN   NNN  EEEEEEEEEEEEE
PPP              HHH         HHH  000         000  NNN      NNNNNN  EEE
PPP              HHH         HHH  000         000  NNN      NNNNNN  EEE
PPP              HHH         HHH  000         000  NNN      NNNNNN  EEE
PPP              HHH         HHH  000         000  NNN         NNN  EEE
PPP              HHH         HHH  000         000  NNN         NNN  EEE
PPP              HHH         HHH  000         000  NNN         NNN  EEE
PPP              HHH         HHH     000000000     NNN         NNN  EEEEEEEEEEEEEEEE
PPP              HHH         HHH     000000000     NNN         NNN  EEEEEEEEEEEEEEEE
PPP              HHH         HHH     000000000     NNN         NNN  EEEEEEEEEEEEEEEE
```

```
_$2

Val
---
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
```

```
**FILE**ID**PHONE
```

```
PPPPPPPP    HH      HH    000000    NN      NN  EEEEEEEEEE
PPPPPPPP    HH      HH    000000    NN      NN  EEEEEEEEEE
PP      PP  HH      HH  00      00  NN      NN  EE
PP      PP  HH      HH  00      00  NN      NN  EE
PP      PP  HH      HH  00      00  NNNN    NN  EE
PP      PP  HH      HH  00      00  NNNN    NN  EE
PPPPPPPP    HHHHHHHHHH  00      00  NN  NN  NN  EEEEEEEE
PPPPPPPP    HHHHHHHHHH  00      00  NN  NN  NN  EEEEEEEE
PP          HH      HH  00      00  NN    NNNN  EE
PP          HH      HH  00      00  NN    NNNN  EE
PP          HH      HH  00      00  NN      NN  EE              ....
PP          HH      HH  00      00  NN      NN  EE              ....
PP          HH      HH    000000    NN      NN  EEEEEEEEEE      ....
PP          HH      HH    000000    NN      NN  EEEEEEEEEE      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
0 %title 'PHONE - VAX/VMS Telephone Facility'
0         module phone (  main=phn$phone,
1                         ident='V04-000') = begin
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1
1 !++
1 ! Facility:     VAX/VMS Telephone Facility, Main Module
1 !
1 ! Abstract:     This facility allows the user to talk to other users on the
1 !               local node or any other node.  The goal of this facility
1 !               is to simulate a real telephone as closely as possible,
1 !               hopefully resulting in good human engineering.
1 !
1 !               Special thanks to Tim Halvorsen, author of TALK, the original
1 !               VAX/VMS telephone facility.
1 !
1 !
1 ! Environment:  Native, User mode.  The following privileges are required for
1 !               full operation; PHONE should be installed with them:
1 !
1 !                       NETMBX  To call over the network.
1 !                       OPER    To ring a phone via broadcasting.
1 !                       PRMMBX  To talk to any other user.
1 !                       WORLD   To obtain information about other processes.
1 !
1 !               For remote communication, we assume all participating nodes
1 !               have DECnet Phase III, with routing.
1 !
1 ! Author: Paul C. Anagnostopoulos, Creation Date: 18 November 1980
1 !
1 ! Modified By:
1 !
1 !       V03-004 BLS0251         Benn Schreiber          8-Dec-1983
```

```
:    58         0058  1 !               Increase size of mailbox name buffer to account for
:    59         0059  1 !               extra overhead characters added.  Convert $TRNLOG to
:    60         0060  1 !               $TRNLNM.
:    61         0061  1 !
:    62         0062  1 !       V03-003 PCA1004         Paul C. Anagnostopoulos  8-Nov-1982
:    63         0063  1 !               $ASSIGN no longer returns SS$_IVDEVNAM when assigning to
:    64         0064  1 !               a mailbox that doesn't exist.  It now returns SS$_NOSUCHDEV.
:    65         0065  1 !
:    66         0066  1 !       V03-002 PCA1000         Paul C. Anagnostopoulos  5-Oct-1982
:    67         0067  1 !               Remove all references to CLI$END_PARSE, which is now obsolete.
:    68         0068  1 !
:    69         0069  1 !       V03-001 PCA0045         Paul Anagnostopoulos    26-Mar-1982
:    70         0070  1 !               Major changes to convert from process name to user name.
:    71         0071  1 !--
```

```
;    73         0072  1 %sbttl 'Module Declarations'
;    74         0073  1 !
;    75         0074  1 ! Libraries and Requires:
;    76         0075  1 !
;    77         0076  1
;    78         0077  1 library 'sys$library:starlet.l32';
;    79         0078  1 literal global_data = 1;              ! To suppress generation of external
;    80         0079  1 require 'phonereq';                   ! data declarations.
;    81         0408  1
;    82         0409  1 !
;    83         0410  1 ! Table of Contents:
;    84         0411  1 !
;    85         0412  1
;    86         0413  1 forward routine
;    87         0414  1         phn$phone: novalue,
;    88         0415  1         phn$init_main: novalue,
;    89         0416  1         phn$dcl_command_line: novalue,
;    90         0417  1         phn$queue_smb: novalue,
;    91         0418  1         phn$kill_smb: novalue,
;    92         0419  1         phn$prepare_users_target: novalue;
;    93         0420  1
;    94         0421  1 !
;    95         0422  1 ! External References:
;    96         0423  1 !
;    97         0424  1
;    98         0425  1 external routine
;    99         0426  1         cli$get_value: addressing_mode(general),
;   100         0427  1         cli$present: addressing_mode(general),
;   101         0428  1         lib$free_vm: addressing_mode(general),
;   102         0429  1         lib$get_vm: addressing_mode(general),
;   103         0430  1         ots$cvt_ti_l: addressing_mode(general),
;   104         0431  1         phn$answered,
;   105         0432  1         phn$busy,
;   106         0433  1         phn$cmd_parse,
;   107         0434  1         phn$exit_handler,
;   108         0435  1         phn$facsimile2,
;   109         0436  1         phn$forced_link,
;   110         0437  1         phn$held,
;   111         0438  1         phn$help2,
;   112         0439  1         phn$hungup,
;   113         0440  1         phn$init_slave,
;   114         0441  1         phn$init_term,
;   115         0442  1         phn$inform,
;   116         0443  1         phn$kbd_get,
;   117         0444  1         phn$kbd_route,
;   118         0445  1         phn$directory2,
;   119         0446  1         phn$listen,
;   120         0447  1         phn$make_pub,
;   121         0448  1         phn$mbx_enable,
;   122         0449  1         phn$mbx_name,
;   123         0450  1         phn$rang_in,
;   124         0451  1         phn$rejected,
;   125         0452  1         phn$ring_out,
;   126         0453  1         phn$talk,
;   127         0454  1         phn$unheld,
;   128         0455  1         str$trim: addressing_mode(general);
;   129         0456  1
```

```
1 !
1 ! Own Variables:
1 !
1
1 ! The following is the head of the SMB queue, which is only manipulated
1 ! by this module.
1
1 own
1         smb_queue_head: vector[2,long]
1                             initial(rep 2 of (smb_queue_head));
```

```
1 !
1 ! Global Variables:
1 !
1 ! The following declarations constitute all the global data for the facility.
1 !
1 ! First we declare a buffer to hold our DECnet node name:
1
1 global
1         own_described_buffer(phn$gq_node_name,9);
1
1 ! Next we declare the variables that will contain the command qualifiers:
1
1 global
1         own_described_buffer(phn$gq_switch_hook,1),
1         phn$gl_viewport_size: long,
1         phn$gb_scroll: byte;
1
1 ! Now we declare the head of the PUB chain.
1
1 global
1         phn$gq_pubhead: vector[2,long]
1                         initial(rep 2 of (phn$gq_pubhead));
1
1 ! Finally, we have a byte of global flags necessary for controlling
1 ! screen dynamics.
1
1 global
1         phn$gb_flags: byte
1                       initial(%b'00000000');
```

```
  1 %sbttl 'PHN$PHONE - Mainline Routine'
  1 !++
  1 ! Functional Description:
  1 !       This is the main routine for the PHONE facility.
  1 !
  1 ! Formal Parameters:
  1 !       none
  1 !
  1 ! Implicit Inputs:
  1 !       global data
  1 !
  1 ! Implicit Outputs:
  1 !       global data
  1 !
  1 ! Returned Value:
  1 !       none
  1 !
  1 ! Side Effects:
  1 !
  1 !--
  1
  1
  2 global routine phn$phone: novalue = begin
  2
  2 own
  2         routine_table: vector[19,long] initial(
  2                 phn$kbd_get,
  2                 phn$kbd_route,
  2                 phn$cmd_parse,
  2                 phn$talk,
  2                 phn$help2,
  2                 phn$ring_out,
  2                 0,                                  ! Network slave SMB only.
  2                 phn$rang_in,
  2                 phn$hungup,
  2                 phn$busy,
  2                 phn$answered,
  2                 phn$rejected,
  2                 0,                                  ! Network slave SMB only.
  2                 phn$listen,
  2                 phn$directory2,
  2                 phn$facsimile2,
  2                 phn$forced_link,
  2                 phn$held,
  2                 phn$unheld);
  2
  2 local
  2         status: long,
  2         trnlnmlst : $itmlst_decl(items=1),
  2         s: ref smb;
  2
  2 ! The very first thing we do is find out if this VAX is running DECnet, and
  2 ! set up a global buffer with the node name, or null string if not.
  2 ! Eliminate any underscore in the name for prettiness.
  2
P 2 $itmlst_init(itmlst=trnlnmlst,
;  228        P 0553  2             (itmcod=lnm$_string,bufadr=.phn$gq_node_name[ptr],
;  229          0554  2                     bufsiz=.phn$gq_node_name[len],retlen=phn$gq_node_name[len]));
;  230          0555  2
;  231        P 0556  2     status = $trnlnm(attr=%ref(lnm$m_case_blind),
;  232        P 0557  2                     tabnam=$descriptor('LNM$SYSTEM'),
;  233        P 0558  2                     lognam=$descriptor('SYS$NODE'),
;  234        P 0559  2                     acmode=%ref(psl$c_exec),
;  235          0560  2                     itmlst=trnlnmlst);
;  236          0561  2
;  237          0562  2     if .status eqlu ss$_nolognam then
;  238          0563  2             phn$gq_node_name[len] = 0
;  239          0564  3     else (
;  240          0565  3             check (.status);
;  241          0566  4             if ch$rchar(.phn$gq_node_name[ptr]) eqlu '_' then (
;  242          0567  4                     dec (phn$gq_node_name[len]);
;  243          0568  4                     inc (phn$gq_node_name[ptr]);
;  244          0569  3             );
;  245          0570  2     );
;  246          0571  2
;  247          0572  2     ! It is possible that we have been fired up as a network slave for
;  248          0573  2     ! someone on a remote node that wants to talk to someone on our node.
;  249          0574  2     ! The following routine will check for this, and will never return
;  250          0575  2     ! if it is the case.
;  251          0576  2
;  252          0577  2     phn$init_slave();
;  253          0578  2
;  254          0579  2     ! I guess not.  Begin an interactive session by initializing.
;  255          0580  2
;  256          0581  2     phn$init_main();
;  257          0582  2
;  258          0583  2     ! This is the main processing loop.  We wait for routines to put Steering
;  259          0584  2     ! Message Blocks on the queue.  We then remove them one by one and call
;  260          0585  2     ! the specified steering message routine, passing it the message text in
;  261          0586  2     ! the block.  Whenever the queue empties, we go back to waiting.
;  262          0587  2
;  263          0588  3     loop (
;  264          0589  3
;  265          0590  3             ! Wait for an SMB to be placed on the queue.  There may already be one.
;  266          0591  3
;  267          0592  3             status = $clref(efn=phn$k_smbefn);
;  268          0593  4             if .status nequ ss$_wasset then (
;  269          0594  4                     check (.status);
;  270          0595  4                     status = $waitfr(efn=phn$k_smbefn);
;  271          0596  4                     check (.status);
;  272          0597  3             );
;  273          0598  3
;  274          0599  3             ! We have at least one SMB.  Go into a loop removing them from the
;  275          0600  3             ! front of the queue.  When we empty the queue, go back and wait.
;  276          0601  3
;  277          0602  4             while not remque(.smb_queue_head[0],s) do (
;  278          0603  4
;  279          0604  4                     ! If the SMB type code is valid, call the appropriate
;  280          0605  4                     ! steering message routine.  Pass it the message text
;  281          0606  4                     ! descriptor, which it can clobber.
;  282          0607  4
;  283          0608  4                     if (.s[smb_w_type] gequ 1) and
;  284          0609  4                        (.s[smb_w_type] lequ %allocation(routine_table)/4) then
4                             (.routine_table[.s[smb_w_type]-1]) (s[smb_q_msg])
4                     else
4                             phn$inform(phn$_badsmb);
4
4                     ! Get rid of the SMB.
4
4                     phn$kill_smb(.s);
3             );
2 );
2
1 end;
```

```
                                                                   .TITLE  PHONE PHONE - VAX/VMS Telephone Facility
                                                                   .IDENT  \V04-000\

                                                                   .PSECT  $PLIT$,NOWRT,NOEXE,2

             4D  45  54  53  59  53  24  4D  4E  4C  00000 P.AAB:  .ASCII  \LNM$SYSTEM\                                  ;
                                                     0000A         .BLKB   2
                                           0000000A  0000C P.AAA:  .LONG   10                                             ;
                                           00000000' 00010         .ADDRESS P.AAB                                          ;
                         45  44  4F  4E  24  53  59  53  00014 P.AAD:  .ASCII  \SYS$NODE\                                  ;
                                           00000008  0001C P.AAC:  .LONG   8                                              ;
                                           00000000' 00020         .ADDRESS P.AAD                                          ;

                                                                   .PSECT  $OWN$,NOEXE,2

                                           00000000' 00000 SMB_QUEUE_HEAD:
                                                                   .ADDRESS SMB_QUEUE_HEAD                                 ;
                                           00000000' 00004         .ADDRESS SMB_QUEUE_HEAD                                 ;
00000000G 00000000G 00000000G 00000000G 00000000G 00000000G 00008 ROUTINE_TABLE:
                                                                   .ADDRESS PHN$KBD_GET, PHN$KBD_ROUTE, -                  ;
                                                                            PHN$CMD_PARSE, PHN$TALK, PHN$HELP2, -          ;
                                                                            PHN$RING_OUT                                   ;
                                           00000000  00020         .LONG   0                                              ;
          00000000G 00000000G 00000000G 00000000G 00000000G 00024         .ADDRESS PHN$RANG_IN, PHN$HUNGUP, PHN$BUSY, -    ;
                                                                            PHN$ANSWERED, PHN$REJECTED                     ;
                                           00000000  00038         .LONG   0                                              ;
00000000G 00000000G 00000000G 00000000G 00000000G 00000000G 0003C         .ADDRESS PHN$LISTEN, PHN$DIRECTORY2, -           ;
                                                                            PHN$FACSIMILE2, PHN$FORCED_LINK, -             ;
                                                                            PHN$HELD, PHN$UNHELD                           ;

                                                                   .PSECT  $GLOBAL$,NOEXE,2

                                           00000009  00000 PHN$GQ_NODE_NAME::
                                                                   .LONG   9                                              ;
                                           00000000' 00004         .ADDRESS PHN$GQ_NODE_NAME+8                            ;
                                                     00008         .BLKB   9
                                                     00011         .BLKB   3
                                           00000001  00014 PHN$GQ_SWITCH_HOOK::
                                                                   .LONG   1                                              ;
                                           00000000' 00018         .ADDRESS PHN$GQ_SWITCH_HOOK+8                          ;
                                                     0001C         .BLKB   1
                                                     0001D         .BLKB   3
                                                     00020 PHN$GL_VIEWPORT_SIZE::
```

```
                                                 .BLKB    4
                            00024 PHN$GB_SCROLL::
                                                 .BLKB    1
                            00025                .BLKB    3
                  00000000' 00028 PHN$GQ_PUBHEAD::
                                                 .ADDRESS PHN$GQ_PUBHEAD                              ;
                  00000000' 0002C                .ADDRESS PHN$GQ_PUBHEAD                              ;
                         00 00030 PHN$GB_FLAGS::
                                                 .BYTE    0                                           ;

                                                 .EXTRN   PHN$_OK, PHN$_ANSWERED
                                                 .EXTRN   PHN$_BUSYCALL, PHN$_CANCALL
                                                 .EXTRN   PHN$_CANTREACH, PHN$_CONFCALL
                                                 .EXTRN   PHN$_DEAD, PHN$_DECNETLINK
                                                 .EXTRN   PHN$_DIRCAN, PHN$_FACSCAN
                                                 .EXTRN   PHN$_HELPCAN, PHN$_HUNGUP
                                                 .EXTRN   PHN$_JUSTRANG, PHN$_LOGGEDOFF
                                                 .EXTRN   PHN$_REJECTED, PHN$_RING
                                                 .EXTRN   PHN$_REJECTJUNK
                                                 .EXTRN   PHN$_SENDINGMAIL
                                                 .EXTRN   PHN$_BADCMD, PHN$_BADHELP
                                                 .EXTRN   PHN$_BADMAILCMD
                                                 .EXTRN   PHN$_BADSMB, PHN$_BADSPEC
                                                 .EXTRN   PHN$_HELPMISSING
                                                 .EXTRN   PHN$_IVREDUNANS
                                                 .EXTRN   PHN$_IVREDUNCALL
                                                 .EXTRN   PHN$_LINKERROR, PHN$_NEEDUSER
                                                 .EXTRN   PHN$_NOCALL, PHN$_NOHOLDS
                                                 .EXTRN   PHN$_NOPORTS, PHN$_NOPRIV
                                                 .EXTRN   PHN$_NOPROC, PHN$_NOTCONV
                                                 .EXTRN   PHN$_ONLYNODE, PHN$_PHONEBUSY
                                                 .EXTRN   PHN$_REMOTEERROR
                                                 .EXTRN   PHN$_TARGTERM, PHN$_UNPLUGGED
                                                 .EXTRN   PHN$_BADTERM, PHN$_SHAREDMBX
                                                 .EXTRN   PHN$_INPUTTERM, CLI$GET_VALUE
                                                 .EXTRN   CLI$PRESENT, LIB$FREE_VM
                                                 .EXTRN   LIB$GET_VM, OTS$CVT_TI_L
                                                 .EXTRN   PHN$ANSWERED, PHN$BUSY
                                                 .EXTRN   PHN$CMD_PARSE, PHN$EXIT_HANDLER
                                                 .EXTRN   PHN$FACSIMILE2, PHN$FORCED_LINK
                                                 .EXTRN   PHN$HELD, PHN$HELP2
                                                 .EXTRN   PHN$HUNGUP, PHN$INIT_SLAVE
                                                 .EXTRN   PHN$INIT_TERM, PHN$INFORM
                                                 .EXTRN   PHN$KBD_GET, PHN$KBD_ROUTE
                                                 .EXTRN   PHN$DIRECTORY2, PHN$LISTEN
                                                 .EXTRN   PHN$MAKE_PUB, PHN$MBX_ENABLE
                                                 .EXTRN   PHN$MBX_NAME, PHN$RANG_IN
                                                 .EXTRN   PHN$REJECTED, PHN$RING_OUT
                                                 .EXTRN   PHN$TALK, PHN$UNHELD
                                                 .EXTRN   STR$TRIM, SYS$TRNLNM
                                                 .EXTRN   SYS$CLREF, SYS$WAITFR

                                                 .PSECT   $CODE$,NOWRT,2

                       003C 00000                .ENTRY   PHN$PHONE, Save R2,R3,R4,R5                 ; 0518
        55 00000000G  00  9E 00002               MOVAB    LIB$SIGNAL, R5                              ;
        54      0000'  CF  9E 00009               MOVAB    PHN$GQ_NODE_NAME, R4                        ;
```

```
                   5E              18  C2 0000E          SUBL2   #24, SP                                         ;
                   50        08    AE  9E 00011          MOVAB   TRNLNMLST, $$ITMBLKPTR                          ; 0554
                   80              64  B0 00015          MOVW    PHN$GQ_NODE_NAME, ($$ITMBLKPTR)+                ;
                   80              02  B0 00018          MOVW    #2, ($$ITMBLKPTR)+                              ;
                   80        04    A4  D0 0001B          MOVL    PHN$GQ_NODE_NAME+4, ($$ITMBLKPTR)+              ;
                   80              64  9E 0001F          MOVAB   PHN$GQ_NODE_NAME, ($$ITMBLKPTR)+                ;
                                   80  D4 00022          CLRL    ($$ITMBLKPTR)+                                  ;
                             08    AE  9F 00024          PUSHAB  TRNLNMLST                                       ; 0560
              08   AE              01  D0 00027          MOVL    #1, 8(SP)                                       ;
                             08    AE  9F 0002B          PUSHAB  8(SP)                                           ;
                          0000'    CF  9F 0002E          PUSHAB  P.AAC                                           ;
                          0000'    CF  9F 00032          PUSHAB  P.AAA                                           ;
              10   AE 02000000     8F  D0 00036          MOVL    #33554432, 16(SP)                               ;
                             10    AE  9F 0003E          PUSHAB  16(SP)                                          ;
       00000000G   00              05  FB 00041          CALLS   #5, SYS$TRNLNM                                  ;
                   53              50  D0 00048          MOVL    R0, STATUS                                      ;
       000001BC    8F              53  D1 0004B          CMPL    STATUS, #444                                    ; 0562
                                   04  12 00052          BNEQ    1$                                              ;
                                   64  B4 00054          CLRW    PHN$GQ_NODE_NAME                                ; 0563
                                   14  11 00056          BRB     3$                                              ;
                   05              53  E8 00058 1$:      BLBS    STATUS, 2$                                      ; 0565
                                   53  DD 0005B          PUSHL   STATUS                                          ;
                   65              01  FB 0005D          CALLS   #1, LIB$SIGNAL                                  ;
              5F   8F        04    B4  91 00060 2$:      CMPB    @PHN$GQ_NODE_NAME+4, #95                        ; 0566
                                   05  12 00065          BNEQ    3$                                              ;
                                   64  B7 00067          DECW    PHN$GQ_NODE_NAME                                ; 0567
                             04    A4  D6 00069          INCL    PHN$GQ_NODE_NAME+4                              ; 0568
           0000G   CF              00  FB 0006C 3$:      CALLS   #0, PHN$INIT_SLAVE                              ; 0577
           0000V   CF              00  FB 00071          CALLS   #0, PHN$INIT_MAIN                               ; 0581
                                   03  DD 00076 4$:      PUSHL   #3                                              ; 0592
       00000000G   00              01  FB 00078          CALLS   #1, SYS$CLREF                                   ;
                   53              50  D0 0007F          MOVL    R0, STATUS                                      ;
                   09              53  D1 00082          CMPL    STATUS, #9                                      ; 0593
                                   1C  13 00085          BEQL    6$                                              ;
                   05              53  E8 00087          BLBS    STATUS, 5$                                      ; 0594
                                   53  DD 0008A          PUSHL   STATUS                                          ;
                   65              01  FB 0008C          CALLS   #1, LIB$SIGNAL                                  ;
                                   03  DD 0008F 5$:      PUSHL   #3                                              ; 0595
       00000000G   00              01  FB 00091          CALLS   #1, SYS$WAITFR                                  ;
                   53              50  D0 00098          MOVL    R0, STATUS                                      ;
                   05              53  E8 0009B          BLBS    STATUS, 6$                                      ; 0596
                                   53  DD 0009E          PUSHL   STATUS                                          ;
                   65              01  FB 000A0          CALLS   #1, LIB$SIGNAL                                  ;
                   52       0000'  DF  0F 000A3 6$:      REMQUE  @SMB_QUEUE_HEAD, S                              ; 0602
                                   CC  1D 000A8          BVS     4$                                              ;
                             0A    A2  B5 000AA          TSTW    10(S)                                           ; 0608
                                   18  13 000AD          BEQL    7$                                              ;
                   13        0A    A2  B1 000AF          CMPW    10(S), #19                                      ; 0609
                                   12  1A 000B3          BGTRU   7$                                              ;
                   50        0A    A2  3C 000B5          MOVZWL  10(S), R0                                       ; 0611
                   50       0000'CF40  D0 000B9          MOVL    ROUTINE_TABLE-4[R0], R0                         ;
                             0C    A2  9F 000BF          PUSHAB  12(S)                                           ;
                   60              01  FB 000C2          CALLS   #1, (R0)                                        ;
                                   08  11 000C5          BRB     8$                                              ;
                        00000000G  8F  DD 000C7 7$:      PUSHL   #PHN$_BADSMB                                    ; 0613
           0000G   CF              01  FB 000CD          CALLS   #1, PHN$INFORM                                  ;
                                   52  DD 000D2 8$:      PUSHL   S                                               ; 0617
```

```
                            0000V  CF             01  FB 000D4          CALLS   #1, PHN$KILL_SMB                                  ;
                                                  C8  11 000D9          BRB     6$                                                ; 0602

; Routine Size:  219 bytes,    Routine Base:  $CODE$ + 0000
```

```
%sbttl 'PHN$INIT_MAIN - Initialize Our Wonderfullness'
!++
! Functional Description:
!       This routine is called at the very beginning of execution
!       to initialize everything.
!
! Formal Parameters:
!       none
!
! Implicit Inputs:
!       global data
!
! Implicit Outputs:
!       global data
!
! Returned Value:
!       none
!
! Side Effects:
!
!--


global routine phn$init_main: novalue = begin

own
        exit_status: long,
        exit_control_block: vector[4,long]
                        initial(0,phn$exit_handler,1,exit_status);
own
        own_described_buffer(user_name,12);
bind
        get_name = uplit(word(12),word(jpi$_username),
                         long(user_name+8),
                         long(user_name),
                         long(0));

local
        status: long,
        op: ref pub;                    ! Pointer to our PUB.


! We begin by obtaining our user name.

status = $getjpi(efn=phn$k_getjpiefn,
                 itmlst=get_name);
check (.status);
status = $waitfr(efn=phn$k_getjpiefn);
check (.status);
str$trim(user_name,user_name,user_name);

! Now we can create a PUB to represent ourselves, known as "our PUB".  Our
! PUB is always assumed to be the first one on the PUB chain.

status = phn$make_pub(user_name,op);
check (.status);
op[pub_v_temporary] = false;
```

```

! Now we have to assign ourselves to our receive mailbox, the mailbox that
! other users will send us messages in.  The mailbox may already have been
! created by some other user trying to call us, so first we try to assign
! to it.

begin
local
        local_described_buffer(mbx_name,4+32);

phn$mbx_name(user_name,mbx_name);
status = $assign(devnam=mbx_name,
                 chan=op[pub_w_channel]);
if (.status nequ ss$_nosuchdev)
    and (.status nequ ss$_ivdevnam) then
        check (.status)
else (
        ! Nope, our mailbox doesn't exist.  Try to create a permanent
        ! mailbox with the name.  Mark it for deletion so we don't leave
        ! crud around later.

        status = $crembx(prmflg=1,
                         chan=op[pub_w_channel],
                         maxmsg=phn$k_mbxsize,
                         lognam=mbx_name);
        if .status nequ ss$_nopriv then (
                check (.status);
                status = $delmbx(chan=.op[pub_w_channel]);
                check (.status);
        ) else (

                ! Too bad, we don't have the privilege to create a permanent
                ! mailbox.  Try to create a temporary one so the guy can at
                ! least talk to him/herself.

                status = $crembx(prmflg=0,
                                 chan=op[pub_w_channel],
                                 maxmsg=phn$k_mbxsize,
                                 lognam=mbx_name);
                check (.status);
        );
);
end;

! Now we can enable our mailbox, allowing an AST to be delivered when someone
! sends us a message.  There may already be a message waiting.

phn$mbx_enable();

! Now we have to obtain the various goodies from the command line.  This is
! done after enabling our mailbox so any messages from other people will be
! handled before any subcommand included on the PHONE command.

phn$dcl_command_line();

! Finally we initialize the user's terminal.  This is done last so that
! anything the user types will be handled after any subcommand included
```

```
;   412          0736  2 ! on the PHONE command.
;   413          0737  2
;   414          0738  2 status = phn$init_term();
;   415          0739  2 check (.status);
;   416          0740  2
;   417          0741  2 ! From this point on, we want to perform some clean-up if the user exits.
;   418          0742  2
;   419          0743  2 status = $dclexh(desblk=exit_control_block);
;   420          0744  2 check (.status);
;   421          0745  2
;   422          0746  2 return;
;   423          0747  2
;   424          0748  1 end;


                                                              .PSECT  $PLIT$,NOWRT,NOEXE,2

                                          000C  00024 P.AAE:  .WORD    12                                  ;
                                          0202  00026         .WORD    514                                 ;
                                     00000000'  00028         .ADDRESS USER_NAME+8                         ;
                                     00000000'  0002C         .ADDRESS USER_NAME                           ;
                                      00000000  00030         .LONG    0                                   ;

                                                              .PSECT  $OWN$,NOEXE,2

                                                00054 EXIT_STATUS:
                                                              .BLKB    4
                                      00000000  00058 EXIT_CONTROL_BLOCK:
                                                              .LONG    0                                   ;
                                     00000000G  0005C         .ADDRESS PHN$EXIT_HANDLER                    ;
                                      00000001  00060         .LONG    1                                   ;
                                     00000000'  00064         .ADDRESS EXIT_STATUS                         ;
                                      0000000C  00068 USER_NAME:
                                                              .LONG    12                                  ;
                                     00000000'  0006C         .ADDRESS USER_NAME+8                         ;
                                                00070         .BLKB    12

                                                      GET_NAME=        P.AAE
                                                              .EXTRN   SYS$GETJPI, SYS$ASSIGN
                                                              .EXTRN   SYS$CREMBX, SYS$DELMBX
                                                              .EXTRN   SYS$DCLEXH

                                                              .PSECT  $CODE$,NOWRT,2

                                          007C  00000         .ENTRY   PHN$INIT_MAIN, Save R2,R3,R4,R5,R6  ; 0645
                       56 00000000G    00   9E  00002         MOVAB    SYS$CREMBX, R6                      ;
                       55      0000'   CF   9E  00009         MOVAB    USER_NAME, R5                       ;
                       54 00000000G    00   9E  0000E         MOVAB    LIB$SIGNAL, R4                      ;
                       5E              30   C2  00015         SUBL2    #48, SP                             ;
                                       7E   7C  00018         CLRQ     -(SP)                               ; 0667
                                       7E   D4  0001A         CLRL     -(SP)                               ;
                            0000'      CF   9F  0001C         PUSHAB   GET_NAME                            ;
                                       7E   7C  00020         CLRQ     -(SP)                               ;
                                       01   DD  00022         PUSHL    #1                                  ;
              00000000G    00          07   FB  00024         CALLS    #7, SYS$GETJPI                      ;
                           53          50   D0  0002B         MOVL     R0, STATUS                          ;
```

```
                        05                53  E8 0002E        BLBS    STATUS, 1$                            ; 0668
                        53                53  DD 00031        PUSHL   STATUS                                ;
                        64                01  FB 00033        CALLS   #1, LIB$SIGNAL                        ;
                                          01  DD 00036  1$:   PUSHL   #1                                    ; 0669
              00000000G 00                01  FB 00038        CALLS   #1, SYS$WAITFR                        ;
                        53                50  D0 0003F        MOVL    R0, STATUS                            ;
                        05                53  E8 00042        BLBS    STATUS, 2$                            ; 0670
                                          53  DD 00045        PUSHL   STATUS                                ;
                        64                01  FB 00047        CALLS   #1, LIB$SIGNAL                        ;
                                          55  DD 0004A  2$:   PUSHL   R5                                    ; 0671
                                          55  DD 0004C        PUSHL   R5                                    ;
                                          55  DD 0004E        PUSHL   R5                                    ;
              00000000G 00                03  FB 00050        CALLS   #3, STR$TRIM                          ;
                                 4020     8F  BB 00057        PUSHR   #^M<R5,SP>                            ; 0676
                  0000G CF                02  FB 0005B        CALLS   #2, PHN$MAKE_PUB                      ;
                        53                50  D0 00060        MOVL    R0, STATUS                            ;
                        05                53  E8 00063        BLBS    STATUS, 3$                            ; 0677
                                          53  DD 00066        PUSHL   STATUS                                ;
                        64                01  FB 00068        CALLS   #1, LIB$SIGNAL                        ;
                        52                6E  D0 0006B  3$:   MOVL    OP, R2                                ; 0678
                  00F0  C2                04  8A 0006E        BICB2   #4, 240(R2)                           ;
                    04  AE                24  D0 00073        MOVL    #36, MBX_NAME                         ; 0687
                    08  AE         0C     AE  9E 00077        MOVAB   MBX_NAME+8, MBX_NAME+4                ;
                                   04     AE  9F 0007C        PUSHAB  MBX_NAME                              ; 0689
                                          55  DD 0007F        PUSHL   R5                                    ;
                  0000G CF                02  FB 00081        CALLS   #2, PHN$MBX_NAME                      ;
                                          7E  7C 00086        CLRQ    -(SP)                                 ; 0691
                        52       00F4     C2  9E 00088        MOVAB   244(R2), R2                           ;
                                          52  DD 0008D        PUSHL   R2                                    ;
                                   10     AE  9F 0008F        PUSHAB  MBX_NAME                              ;
              00000000G 00                04  FB 00092        CALLS   #4, SYS$ASSIGN                        ;
                        53                50  D0 00099        MOVL    R0, STATUS                            ;
              00000908  8F                53  D1 0009C        CMPL    STATUS, #2312                         ; 0692
                                          09  13 000A3        BEQL    4$                                    ;
              00000144  8F                53  D1 000A5        CMPL    STATUS, #324                          ; 0693
                                          45  12 000AC        BNEQ    8$                                    ;
                                   04     AE  9F 000AE  4$:   PUSHAB  MBX_NAME                              ; 0703
                                          7E  7C 000B1        CLRQ    -(SP)                                 ;
                                          7E  D4 000B3        CLRL    -(SP)                                 ;
                        7E       0100     8F  3C 000B5        MOVZWL  #256, -(SP)                           ;
                                          52  DD 000BA        PUSHL   R2                                    ;
                                          01  DD 000BC        PUSHL   #1                                    ;
                        66                07  FB 000BE        CALLS   #7, SYS$CREMBX                        ;
                        53                50  D0 000C1        MOVL    R0, STATUS                            ;
                        24                53  D1 000C4        CMPL    STATUS, #36                           ; 0704
                                          14  13 000C7        BEQL    6$                                    ;
                        05                53  E8 000C9        BLBS    STATUS, 5$                            ; 0705
                                          53  DD 000CC        PUSHL   STATUS                                ;
                        64                01  FB 000CE        CALLS   #1, LIB$SIGNAL                        ;
                        7E                62  3C 000D1  5$:   MOVZWL  (R2), -(SP)                           ; 0706
              00000000G 00                01  FB 000D4        CALLS   #1, SYS$DELMBX                        ;
                                          13  11 000DB        BRB     7$                                    ;
                                   04     AE  9F 000DD  6$:   PUSHAB  MBX_NAME                              ; 0717
                                          7E  7C 000E0        CLRQ    -(SP)                                 ;
                                          7E  D4 000E2        CLRL    -(SP)                                 ;
                        7E       0100     8F  3C 000E4        MOVZWL  #256, -(SP)                           ;
                                          52  DD 000E9        PUSHL   R2                                    ;
                                 7E  D4 000EB          CLRL    -(SP)                                   ;
                    66           07  FB 000ED          CALLS   #7, SYS$CREMBX                          ;
                    53           50  D0 000F0 7$:      MOVL    R0, STATUS                              ;
                    05           53  E8 000F3 8$:      BLBS    STATUS, 9$                              ; 0718
                                 53  DD 000F6          PUSHL   STATUS                                  ;
                    64           01  FB 000F8          CALLS   #1, LIB$SIGNAL                          ;
              0000G CF           00  FB 000FB 9$:      CALLS   #0, PHN$MBX_ENABLE                      ; 0726
              0000V CF           00  FB 00100          CALLS   #0, PHN$DCL_COMMAND_LINE                ; 0732
              0000G CF           00  FB 00105          CALLS   #0, PHN$INIT_TERM                       ; 0738
                    53           50  D0 0010A          MOVL    R0, STATUS                              ;
                    05           53  E8 0010D          BLBS    STATUS, 10$                             ; 0739
                                 53  DD 00110          PUSHL   STATUS                                  ;
                    64           01  FB 00112          CALLS   #1, LIB$SIGNAL                          ;
                           F0    A5  9F 00115 10$:     PUSHAB  EXIT_CONTROL_BLOCK                      ; 0743
         00000000G  00           01  FB 00118          CALLS   #1, SYS$DCLEXH                          ;
                    53           50  D0 0011F          MOVL    R0, STATUS                              ;
                    05           53  E8 00122          BLBS    STATUS, 11$                             ; 0744
                                 53  DD 00125          PUSHL   STATUS                                  ;
                    64           01  FB 00127          CALLS   #1, LIB$SIGNAL                          ;
                                     04 0012A 11$:     RET                                             ; 0748

; Routine Size:  299 bytes,    Routine Base:  $CODE$ + 00DB
```

```
1 %sbttl 'PHN$DCL_COMMAND_LINE - Get Command Line Goodies'
1 !++
1 ! Functional Description:
1 !       This routine is called to "parse the command line".  All the work
1 !       has been done by DCL, but we need to use the callback mechanism
1 !       to get the information.
1 !
1 ! Formal Parameters:
1 !       none
1 !
1 ! Implicit Inputs:
1 !       global data
1 !
1 ! Implicit Outputs:
1 !       global data
1 !
1 ! Returned Value:
1 !       none
1 !
1 ! Side Effects:
1 !
1 !--
1
1
2 global routine phn$dcl_command_line: novalue = begin
2
2 local
2       status: long;
2
2
2 ! First we get the switch hook character as specified by the qualifier.
2
2 cli$get_value(describe('SWITCH_HOOK'),phn$gq_switch_hook);
2
2 ! Now we get the viewport size, as specified by the qualifier.  We have
2 ! to convert the value to binary and make sure it's within the valid range.
2
3 begin
3 local
3       local_described_buffer(size_buf,10);
3
3 cli$get_value(describe('VIEWPORT_SIZE'),size_buf);
3 status = ots$cvt_ti_l(size_buf,phn$gl_viewport_size,4,%b'11');
3 if .status eqlu ss$_normal then
3       phn$gl_viewport_size = min(max(pub_k_minlines,.phn$gl_viewport_size),
3                                  pub_k_maxlines)
3 else
3       phn$gl_viewport_size = pub_k_maxlines;
2 end;
2
2 ! Now we get the scroll flag, as specified by the qualifier.
2
2 phn$gb_scroll = cli$present(describe('SCROLL'));
2
2 ! Now we have to retrieve the command line tokens.  If present, these
2 ! make up a phone command that we are to execute first.  Sit in a loop
2 ! and queue cmd_parse steering messages for each token.
```

```
2
3 incru i from 1 to 4 do (
3
3           local
3                   local_described_buffer(token_buf,80);
3
4           cli$get_value((case .i from 1 to 4 of set
4                           [1]:    describe('TOKEN1');
4                           [2]:    describe('TOKEN2');
4                           [3]:    describe('TOKEN3');
4                           [4]:    describe('TOKEN4');
3                           tes),                           token_buf);
3           str$trim(token_buf,token_buf,token_buf);
3
4           if .token_buf[len] gequ 1 then (
4                   phn$queue_smb(smb__cmd_parse,describe(' '));
4                   phn$queue_smb(smb__cmd_parse,token_buf);
3           );
2 );
2
2 ! Now we can queue a carriage return to end the command.  It doesn't hurt
2 ! if there wasn't a command.
2
2 phn$queue_smb(smb__cmd_parse,describe(%char(ret)));
2
2 return;
2
1 end;



                                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

          4B  4F  4F  48  5F  48  43  54  49  57  53  00034 P.AAG:  .ASCII  \SWITCH_HOOK\                        ;
                                                      0003F         .BLKB   1
                                           0000000B   00040 P.AAF:  .LONG   11                                   ;
                                           00000000'  00044         .ADDRESS P.AAG                               ;
  45  5A  49  53  5F  54  52  4F  50  57  45  49  56  00048 P.AAI:  .ASCII  \VIEWPORT_SIZE\                      ;
                                                      00055         .BLKB   3
                                           0000000D   00058 P.AAH:  .LONG   13                                   ;
                                           00000000'  0005C         .ADDRESS P.AAI                               ;
                          4C  4C  4F  52  43  53      00060 P.AAK:  .ASCII  \SCROLL\                             ;
                                                      00066         .BLKB   2
                                           00000006   00068 P.AAJ:  .LONG   6                                    ;
                                           00000000'  0006C         .ADDRESS P.AAK                               ;
                          31  4E  45  4B  4F  54      00070 P.AAM:  .ASCII  \TOKEN1\                             ;
                                                      00076         .BLKB   2
                                           00000006   00078 P.AAL:  .LONG   6                                    ;
                                           00000000'  0007C         .ADDRESS P.AAM                               ;
                          32  4E  45  4B  4F  54      00080 P.AAO:  .ASCII  \TOKEN2\                             ;
                                                      00086         .BLKB   2
                                           00000006   00088 P.AAN:  .LONG   6                                    ;
                                           00000000'  0008C         .ADDRESS P.AAO                               ;
                          33  4E  45  4B  4F  54      00090 P.AAQ:  .ASCII  \TOKEN3\                             ;
                                                      00096         .BLKB   2
                                           00000006   00098 P.AAP:  .LONG   6                                    ;
                                           00000000'  0009C         .ADDRESS P.AAQ                               ;
```

```
                        34  4E  45  4B  4F  54  000A0 P.AAS:   .ASCII   \TOKEN4\                                     ;
                                                000A6          .BLKB    2
                                     00000006   000A8 P.AAR:   .LONG    6                                            ;
                                     00000000'  000AC          .ADDRESS P.AAS                                        ;
                                           20   000B0 P.AAU:   .ASCII   \ \                                          ;
                                                000B1          .BLKB    3
                                     00000001   000B4 P.AAT:   .LONG    1                                            ;
                                     00000000'  000B8          .ADDRESS P.AAU                                        ;
                                           0D   000BC P.AAW:   .ASCII   <13>                                         ;
                                                000BD          .BLKB    3
                                     00000001   000C0 P.AAV:   .LONG    1                                            ;
                                     00000000'  000C4          .ADDRESS P.AAW                                        ;



                                                               .PSECT   $CODE$,NOWRT,2

                                         007C   00000          .ENTRY   PHN$DCL_COMMAND_LINE, Save R2,R3,R4,R5,R6    ; 0773
                        56      0000V  CF  9E   00002          MOVAB    PHN$QUEUE_SMB, R6                            ;
                        55      0000'  CF  9E   00007          MOVAB    PHN$GL_VIEWPORT_SIZE, R5                     ;
                        54  00000000G  00  9E   0000C          MOVAB    CLI$GET_VALUE, R4                            ;
                        53      0000'  CF  9E   00013          MOVAB    P.AAF, R3                                    ;
                        5E         A8  AE  9E   00018          MOVAB    -88(SP), SP                                  ;
                                   F4  A5  9F   0001C          PUSHAB   PHN$GQ_SWITCH_HOOK                           ; 0781
                                       53  DD   0001F          PUSHL    R3                                           ;
                        64             02  FB   00021          CALLS    #2, CLI$GET_VALUE                            ;
                44      AE             0A  D0   00024          MOVL     #10, SIZE_BUF                                ; 0788
                48      AE         4C  AE  9E   00028          MOVAB    SIZE_BUF+8, SIZE_BUF+4                       ;
                                   44  AE  9F   0002D          PUSHAB   SIZE_BUF                                     ; 0790
                                   18  A3  9F   00030          PUSHAB   P.AAR                                        ;
                        64             02  FB   00033          CALLS    #2, CLI$GET_VALUE                            ;
                                       03  DD   00036          PUSHL    #3                                           ; 0791
                                       04  DD   00038          PUSHL    #4                                           ;
                                       55  DD   0003A          PUSHL    R5                                           ;
                                   50  AE  9F   0003C          PUSHAB   SIZE_BUF                                     ;
          00000000G     00             04  FB   0003F          CALLS    #4, OTS$CVT_TI_L                             ;
                        01             50  D1   00046          CMPL     STATUS, #1                                   ; 0792
                                   18  12       00049          BNEQ     3$                                           ;
                        50             65  D0   0004B          MOVL     PHN$GL_VIEWPORT_SIZE, R0                     ; 0793
                        03             50  D1   0004E          CMPL     R0, #3                                       ;
                                       03  18   00051          BGEQ     1$                                           ;
                        50             03  D0   00053          MOVL     #3, R0                                       ;
                        0A             50  D1   00056 1$:      CMPL     R0, #10                                      ;
                                       03  15   00059          BLEQ     2$                                           ;
                        50             0A  D0   0005B          MOVL     #10, R0                                      ;
                        65             50  D0   0005E 2$:      MOVL     R0, PHN$GL_VIEWPORT_SIZE                     ;
                                       03  11   00061          BRB      4$                                           ;
                        65             0A  D0   00063 3$:      MOVL     #10, PHN$GL_VIEWPORT_SIZE                    ; 0796
                                   28  A3  9F   00066 4$:      PUSHAB   P.AAJ                                        ; 0801
          00000000G     00             01  FB   00069          CALLS    #1, CLI$PRESENT                              ;
                04      A5             50  90   00070          MOVB     R0, PHN$GB_SCROLL                            ;
                        52             01  D0   00074          MOVL     #1, I                                        ; 0807
                        6E         50  8F  9A   00077 5$:      MOVZBL   #80, TOKEN_BUF                               ; 0810
                04      AE         08  AE  9E   0007B          MOVAB    TOKEN_BUF+8, TOKEN_BUF+4                     ;
                                       5E  DD   00080          PUSHL    SP                                           ; 0812
                03      01             52  CF   00082          CASEL    I, #1, #3                                    ;
   001A         0014    000E       0008         00086 6$:      .WORD    7$-6$,-                                      ;
```

```
                                                                  8$-6$,-                                 ;
                                                                  9$-6$,-                                 ;
                                                                  10$-6$                                  ;
                    50      38  A3  9E 0008E 7$:      MOVAB   P.AAL, R0                               ; 0813
                                10  11 00092          BRB     11$                                     ;
                    50      48  A3  9E 00094 8$:      MOVAB   P.AAN, R0                               ; 0814
                                0A  11 00098          BRB     11$                                     ;
                    50      58  A3  9E 0009A 9$:      MOVAB   P.AAP, R0                               ; 0815
                                04  11 0009E          BRB     11$                                     ;
                    50      68  A3  9E 000A0 10$:     MOVAB   P.AAR, R0                               ; 0816
                                50  DD 000A4 11$:     PUSHL   R0                                      ;
                    64          02  FB 000A6          CALLS   #2, CLI$GET_VALUE                       ; 0812
                                5E  DD 000A9          PUSHL   SP                                      ; 0818
                            04  AE  9F 000AB          PUSHAB  TOKEN_BUF                               ;
                            08  AE  9F 000AE          PUSHAB  TOKEN_BUF                               ;
          00000000G 00          03  FB 000B1          CALLS   #3, STR$TRIM                            ;
                                6E  B5 000B8          TSTW    TOKEN_BUF                               ; 0820
                                0F  13 000BA          BEQL    12$                                     ;
                            74  A3  9F 000BC          PUSHAB  P.AAT                                   ; 0821
                                03  DD 000BF          PUSHL   #3                                      ;
                    66          02  FB 000C1          CALLS   #2, PHN$QUEUE_SMB                       ;
                                5E  DD 000C4          PUSHL   SP                                      ; 0822
                                03  DD 000C6          PUSHL   #3                                      ;
                    66          02  FB 000C8          CALLS   #2, PHN$QUEUE_SMB                       ;
                                52  D6 000CB 12$:     INCL    I                                       ; 0807
                    04          52  D1 000CD          CMPL    I, #4                                   ;
                                A5  1B 000D0          BLEQU   5$                                      ;
                          0080  C3  9F 000D2          PUSHAB  P.AAV                                   ; 0829
                                03  DD 000D6          PUSHL   #3                                      ;
                    66          02  FB 000D8          CALLS   #2, PHN$QUEUE_SMB                       ;
                                    04 000DB          RET                                             ; 0833

; Routine Size:  220 bytes,    Routine Base:  $CODE$ + 0206
```

```
1 %sbttl 'PHN$QUEUE_SMB - Queue a Steering Message Block'
1 !++
1 ! Functional Description:
1 !       This routine is called to create a Steering Message Block and
1 !       queue in onto the SMB queue.
1 !
1 ! Formal Parameters:
1 !       type            The type code for the SMB.
1 !       text            Address of descriptor of optional message text.
1 !
1 ! Implicit Inputs:
1 !       global data
1 !
1 ! Implicit Outputs:
1 !       global data
1 !
1 ! Returned Value:
1 !       none
1 !
1 ! Side Effects:
1 !
1 !--
1
1
2 global routine phn$queue_smb(type,text): novalue = begin
2
2 bind
2       text_dsc = .text: descriptor;
2
2 local
2       status: long,
2       smb_size: long,
2       s: ref smb;
2
2 builtin
2       nullparameter;
2
2
2 ! We begin by calculating the total size of the SMB and allocating
2 ! memory for it.
2
3 smb_size = smb_k_size + (if nullparameter(2) then 0
2                                           else .text_dsc[len]);
2 status = lib$get_vm(smb_size,s);
2 check (.status);
2
2 ! Now we fill in the new SMB with its length, the message type code,
2 ! and the message text.  We also build a descriptor for the message text.
2
2 s[smb_w_length] = .smb_size;
2 s[smb_w_type] = .type;
3 begin
3 bind
3       smb_msg_dsc = s[smb_q_msg]: descriptor;
3
3 if nullparameter(2) then
3       smb_msg_dsc[len] = 0
```

```
;  569           0891  4 else (
;  570           0892  4         ch$move(.text_dsc[len],.text_dsc[ptr],s[smb_t_msgbuf]);
;  571           0893  4         smb_msg_dsc[len] = .text_dsc[len];
;  572           0894  4         smb_msg_dsc[ptr] = s[smb_t_msgbuf];
;  573           0895  3 );
;  574           0896  2 end;
;  575           0897  2
;  576           0898  2 ! Now we can queue the new SMB onto the end of the steering message queue.
;  577           0899  2 ! If the queue was empty, make sure we set the event flag to awaken the
;  578           0900  2 ! main loop.
;  579           0901  2
;  580           0902  2 status = insque(.s,.smb_queue_head[1]);
;  581           0903  2 if .status eqlu 1 then
;  582           0904  2         $setef(efn=phn$k_smbefn);
;  583           0905  2
;  584           0906  2 return;
;  585           0907  2
;  586           0908  1 end;


                                                          .EXTRN  SYS$SETEF

                                      03FC 00000          .ENTRY  PHN$QUEUE_SMB, Save R2,R3,R4,R5,R6,R7,R8,R9 ; 0858
                   5E              08   C2 00002          SUBL2   #8, SP                                     ;
                   58        08    AC   D0 00005          MOVL    TEXT, R8                                   ; 0861
                   02              6C   91 00009          CMPB    (AP), #2                                   ; 0875
                                   05   1F 0000C          BLSSU   1$                                         ;
                             08    AC   D5 0000E          TSTL    8(AP)                                      ;
                                   04   12 00011          BNEQ    2$                                         ;
                                   50   D4 00013 1$:      CLRL    R0                                         ;
                                   03   11 00015          BRB     3$                                         ;
                   50              68   3C 00017 2$:      MOVZWL  (R8), R0                                   ; 0876
            04     AE        14    A0   9E 0001A 3$:      MOVAB   20(R0), SMB_SIZE                           ; 0875
                                   5E   DD 0001F          PUSHL   SP                                         ; 0877
                             08    AE   9F 00021          PUSHAB  SMB_SIZE                                   ;
     00000000G     00              02   FB 00024          CALLS   #2, LIB$GET_VM                             ;
                   59              50   D0 0002B          MOVL    R0, STATUS                                 ;
                   09              59   E8 0002E          BLBS    STATUS, 4$                                 ; 0878
                                   59   DD 00031          PUSHL   STATUS                                     ;
     00000000G     00              01   FB 00033          CALLS   #1, LIB$SIGNAL                             ;
                   57              6E   D0 0003A 4$:      MOVL    S, R7                                      ; 0883
            08     A7        04    AE   B0 0003D          MOVW    SMB_SIZE, 8(R7)                            ;
            0A     A7        04    AC   B0 00042          MOVW    TYPE, 10(R7)                               ; 0884
                   56        0C    A7   9E 00047          MOVAB   12(R7), R6                                 ; 0887
                   02              6C   91 0004B          CMPB    (AP), #2                                   ; 0889
                                   05   1F 0004E          BLSSU   5$                                         ;
                             08    AC   D5 00050          TSTL    8(AP)                                      ;
                                   04   12 00053          BNEQ    6$                                         ;
                                   66   B4 00055 5$:      CLRW    (R6)                                       ; 0890
                                   0E   11 00057          BRB     7$                                         ;
  14     A7        04      B8      68   28 00059 6$:      MOVC3   (R8), @4(R8), 20(R7)                       ; 0892
                   66              68   B0 0005F          MOVW    (R8), (R6)                                 ; 0893
            04     A6        14    A7   9E 00062          MOVAB   20(R7), 4(R6)                              ; 0894
                                   50   D4 00067 7$:      CLRL    R0                                         ; 0902
            0000'  DF              67   0E 00069          INSQUE  (R7), @SMB_QUEUE_HEAD+4                    ;
                                   02   12 0006E          BNEQ    8$                                         ;
```

```
                                       50  D6 00070          INCL    R0                                          ;
                          59           50  D0 00072 8$:      MOVL    R0, STATUS                                  ;
                          01           59  D1 00075          CMPL    STATUS, #1                                  ; 0903
                                       09  12 00078          BNEQ    9$                                          ;
                                       03  DD 0007A          PUSHL   #3                                          ; 0904
                  00000000G  00        01  FB 0007C          CALLS   #1, SYS$SETEF                               ;
                                           04 00083 9$:      RET                                                 ; 0908

; Routine Size:  132 bytes,    Routine Base:  $CODE$ + 02E2
;  588          0909  1 %sbttl 'PHN$KILL_SMB - Kill an Obsolete SMB'
;  589          0910  1 !++
;  590          0911  1 ! Functional Description:
;  591          0912  1 !       This routine is called to deallocate an obsolete Steering
;  592          0913  1 !       Message Block.
;  593          0914  1 !
;  594          0915  1 ! Formal Parameters:
;  595          0916  1 !       obsolete_smb    Address of the obsolete SMB.
;  596          0917  1 !
;  597          0918  1 ! Implicit Inputs:
;  598          0919  1 !       global data
;  599          0920  1 !
;  600          0921  1 ! Implicit Outputs:
;  601          0922  1 !       global data
;  602          0923  1 !
;  603          0924  1 ! Returned Value:
;  604          0925  1 !       none
;  605          0926  1 !
;  606          0927  1 ! Side Effects:
;  607          0928  1 !
;  608          0929  1 !--
;  609          0930  1
;  610          0931  1
;  611          0932  2 global routine phn$kill_smb(obsolete_smb): novalue = begin
;  612          0933  2
;  613          0934  2 bind
;  614          0935  2         os = .obsolete_smb: smb;
;  615          0936  2
;  616          0937  2 local
;  617          0938  2         status: long,
;  618          0939  2         smb_size: long;
;  619          0940  2
;  620          0941  2
;  621          0942  2 ! All we have to do is get the length of the SMB and deallocate it.
;  622          0943  2
;  623          0944  2 smb_size = .os[smb_w_length];
;  624          0945  2 status = lib$free_vm(smb_size,obsolete_smb);
;  625          0946  2 check (.status);
;  626          0947  2 return;
;  627          0948  2
;  628          0949  1 end;



                                            0000 00000          .ENTRY  PHN$KILL_SMB, Save nothing                          ; 0932
                    50         04   AC   D0 00002          MOVL    OBSOLETE_SMB, R0                                     ; 0935
                    7E         08   A0   3C 00006          MOVZWL  8(R0), SMB_SIZE                                      ; 0944
                               04   AC   9F 0000A          PUSHAB  OBSOLETE_SMB                                         ; 0945
                               04   AE   9F 0000D          PUSHAB  SMB_SIZE                                             ;
          00000000G 00              02   FB 00010          CALLS   #2, LIB$FREE_VM                                      ;
                    09              50   E8 00017          BLBS    STATUS, 1$                                           ; 0946
                                    50   DD 0001A          PUSHL   STATUS                                               ;
          00000000G 00              01   FB 0001C          CALLS   #1, LIB$SIGNAL                                       ;
                                         04 00023 1$:      RET                                                          ; 0949
```

```
; Routine Size:  36 bytes,    Routine Base:  $CODE$ + 0366
```

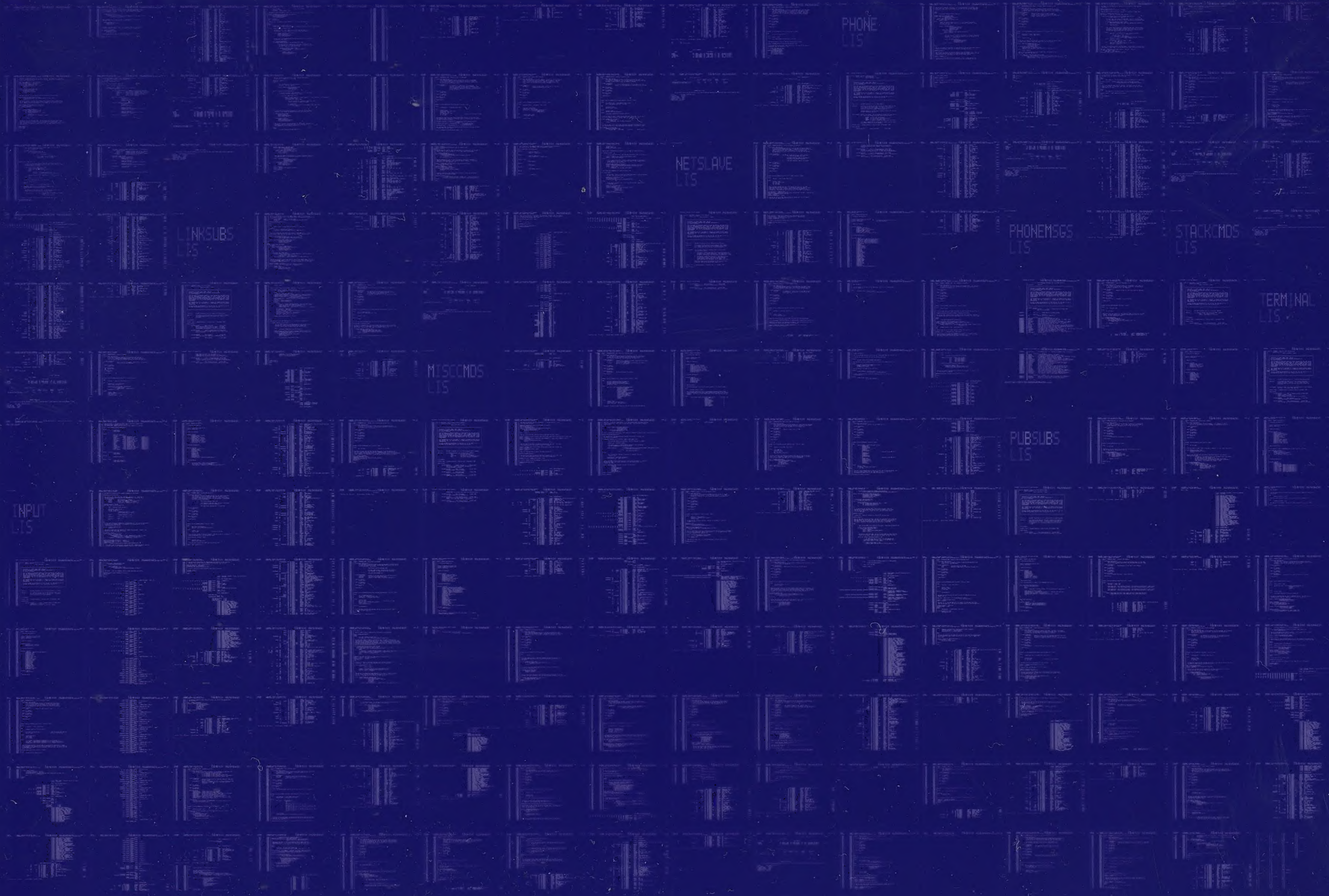
```
  1 %sbttl 'ANL$PREPARE_USERS_TARGET - Prepare Target Entered by User'
  1 !++
  1 ! Functional Description:
  1 !       This routine is called to prepare a target entered by the user.
  1 !       It performs a logical name translation in case the user entered
  1 !       a logical name.  It also handles node names without the double colon.
  1 !
  1 ! Formal Parameters:
  1 !       users           Address of descriptor of user's target.
  1 !       just_node       A flag, true if target is just a node name.
  1 !       final           Address of descriptor of buffer to receive final
  1 !                       target.  We set the length.
  1 !
  1 ! Implicit Inputs:
  1 !       global data
  1 !
  1 ! Implicit Outputs:
  1 !       global data
  1 !
  1 ! Returned Value:
  1 !       none
  1 !
  1 ! Side Effects:
  1 !
  1 !--
  1
  1
  2 global routine phn$prepare_users_target(users,just_node,final): novalue = begin
  2
  2 bind
  2         users_dsc = .users: descriptor,
  2         final_dsc = .final: descriptor;
  2
  2 local
  2         status: long;
  2
  2
  2 ! First we try translating the user's target as if it were a logical name.
  2 ! If that fails, we just move the target into the final buffer.
  2
P 2 status = $trnlog(lognam=users_dsc,
P 2                         rsllen=final_dsc[len],
  2                         rslbuf=final_dsc);
  3 if .status eqlu ss$_ivlognam then (
  3         final_dsc[len] = .users_dsc[len];
  3         ch$move(.users_dsc[len],.users_dsc[ptr], .final_dsc[ptr]);
  2 ) else
  2         check (.status);
  2
  2 ! Now if this is supposed to be just a node name, the double colon is
  2 ! optional.  Add it if we need to.
  2
  2 if .just_node and .final_dsc[len] gtru 0 then
  3         if ch$neq(2,.final_dsc[ptr]+.final_dsc[len]-2, 2,uplit byte ('::'),' ') then (
  3                 ch$move(2,uplit byte ('::'), .final_dsc[ptr]+.final_dsc[len]);
  3                 final_dsc[len] = .final_dsc[len] + 2;
  2         );
```

```
;  687          1007  2
;  688          1008  2 return;
;  689          1009  2
;  690          1010  1 end;


                                                              .PSECT  $PLIT$,NOWRT,NOEXE,2

                                 3A  3A  000C8 P.AAX:   .ASCII  \::\                                              ;
                                 3A  3A  000CA P.AAY:   .ASCII  \::\                                              ;

                                                        .EXTRN  SYS$TRNLOG

                                                        .PSECT  $CODE$,NOWRT,2

                                    007C 00000          .ENTRY  PHN$PREPARE_USERS_TARGET, Save R2,R3,R4,R5,-; 0977
                                                                R6                                                ;
                     52         04   AC  D0 00002       MOVL    USERS, R2                                         ; 0980
                     56         0C   AC  D0 00006       MOVL    FINAL, R6                                         ; 0981
                                     7E  7C 0000A       CLRQ    -(SP)                                             ; 0992
                                     7E  D4 0000C       CLRL    -(SP)                                             ;
                                     56  DD 0000E       PUSHL   R6                                                ;
                              0044   8F  BB 00010       PUSHR   #^M<R2,R6>                                        ;
          00000000G  00              06  FB 00014       CALLS   #6, SYS$TRNLOG                                    ;
          00000154   8F              50  D1 0001B       CMPL    STATUS, #340                                      ; 0993
                                     0B  12 00022       BNEQ    1$                                                ;
                     66              62  B0 00024       MOVW    (R2), (R6)                                        ; 0994
      04    B6      04   B2          62  28 00027       MOVC3   (R2), @4(R2), @4(R6)                              ; 0995
                                     0C  11 0002D       BRB     2$                                                ; 0993
                     09              50  E8 0002F 1$:   BLBS    STATUS, 2$                                        ; 0997
                                     50  DD 00032       PUSHL   STATUS                                            ;
          00000000G  00              01  FB 00034       CALLS   #1, LIB$SIGNAL                                    ;
                     1B         08   AC  E9 0003B 2$:   BLBC    JUST_NODE, 3$                                     ; 1002
                                     66  B5 0003F       TSTW    (R6)                                              ;
                                     17  13 00041       BEQL    3$                                                ;
                     50              66  3C 00043       MOVZWL  (R6), R0                                          ; 1003
                     50         04   A6  C0 00046       ADDL2   4(R6), R0                                         ;
               0000' CF         FE   A0  B1 0004A       CMPW    -2(R0), P.AAX                                     ;
                                     08  13 00050       BEQL    3$                                                ;
                     60      0000'   CF  B0 00052       MOVW    P.AAY, (R0)                                       ; 1004
                     66              02  A0 00057       ADDW2   #2, (R6)                                          ; 1005
                                         04 0005A 3$:   RET                                                       ; 1010

; Routine Size:  91 bytes,    Routine Base:  $CODE$ + 038A


;  691          1011  1
;  692          1012  0 end eludom



                                                        .EXTRN  LIB$SIGNAL

;                                 PSECT SUMMARY
;
```

```
;       Name                         Bytes                          Attributes
;
;  $OWN$                               124  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $GLOBAL$                             49  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                              204  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                              997  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)




;                             Library Statistics
;
;                                                 -------- Symbols --------       Pages        Processing
;       File                                      Total    Loaded   Percent       Mapped       Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1               9776        35         0        581           00:00.7




;                                    COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:PHONE/OBJ=OBJ$:PHONE MSRC$:PHONE/UPDATE=(ENH$:PHONE)

; Size:           997 code + 377 data bytes
; Run Time:          00:14.9
; Elapsed Time:      00:59.4
; Lines/CPU Min:      4077
; Lexemes/CPU-Min: 33215
; Memory Used:  156 pages
; Compilation Complete
```

0305 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY


PHONE
LIS

NETSLAVE
LIS

LINKSUBS
LIS

PHONEMSGS
LIS

STACKCMDS
LIS

TERMINAL
LIS

MISCCMDS
LIS

PUBSUBS
LIS

INPUT
LIS